Групповой проект по базам данных

Пояснительная записка

Участники команды:

Карабаш Радимир БПИ195

Кенесбек Ерасыл БПИ195

Мурзабеков Султан БПИ195

1. **Предназначение разрабатываемой программы**

   База данных, которая в дальнейшем будет рассматриваться в данной пояснительной записке, используется в мобильном приложение по привлечению инвесторов в стартап на этапе краудфайдинга.

   Существуют 2 типа пользователей:
   - Инвестор: получает информацию о стартапе, инвестирует в стартап;
   - Стартапер: создает стартап, привлекает инвестиции, может редактировать стартап.

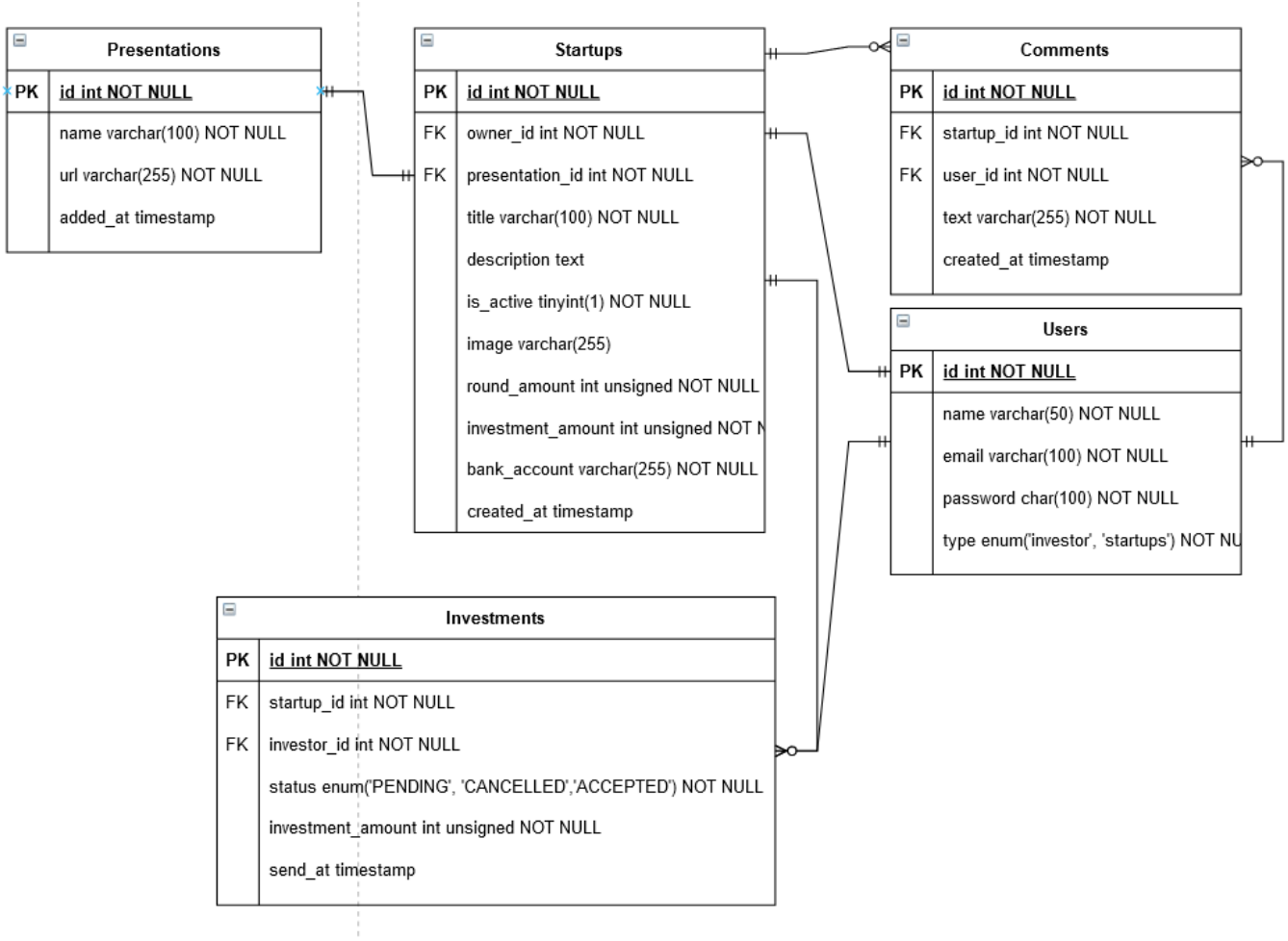
2. **Функциональные требования**
   - Хранение информации о стартапе: название, описание, дата создания, сумма необходимых инвестиций, сумма привлеченных инвестиций, кол-во инвесторов, изображение стартапа, файл с презентацией бизнес-плана, банковский счет;
   - Хранение информации о инвестициях: сумма инвестиций, дата создания инвестиций, статус передачи инвестиции;
   - Хранение информации о пользователе: имя пользователя, email(логин), пароль, тип пользователя.
   - Хранение информации о комментариях: текст комментария, дата создания комментария;
   - Хранение информации о файлах презентаций: имя файла, url файла, дата добавления файла;
   - CRUD-операции для стартапов;
   - CRUD-операции для пользователей;
   - Получение всех инвестиций инвестора;
   - Получение всех стартапов пользователя;
   - Получение всех инвестиций стартапа;
   - Проверка роли пользователя;
   - Получение всех инвестиций по статусу передачи;
   - Активация/деактивация стартапа;

- Получение пользователей в зависимости от их типа.

3. Нефункциональные требования

- Удаление пользователя невозможно при активном стартапе;
- Удаление и деактивация стартапа при частичном сборе необходимых инвестиций невозможно;
- При удалении пользователя, у которого имеется стартап, собравший необходимые инвестиции, данный стартап деактивируется и далее не отображается;
- Инвестиция не может существовать без привязки к инвестору и стартапу;
- Комментарий не может существовать без привязки к пользователю и стартапу.

4. Предварительная схема БД



5. Ограничения на данные

- Один стартап должен принадлежать только к одному стартаперу, и может иметь 0 или больше инвесторов;
- Одна инвестиция может принадлежать только к одному инвестору;
- Комментарии может принадлежать только к одному стартапу, и к одному пользователю;
- Презентация бизнес-плана может принадлежать только к одному стартапу;

- Пользователи могут быть только 2 типов;
- Может существовать только 3 статуса инвестиции;
- Одна инвестиция может принадлежать только к одному стартапу.

6. Нормализация

Проверим нашу предварительную схему на соответствие со следующими нормальными формами:

Первая нормальная форма:

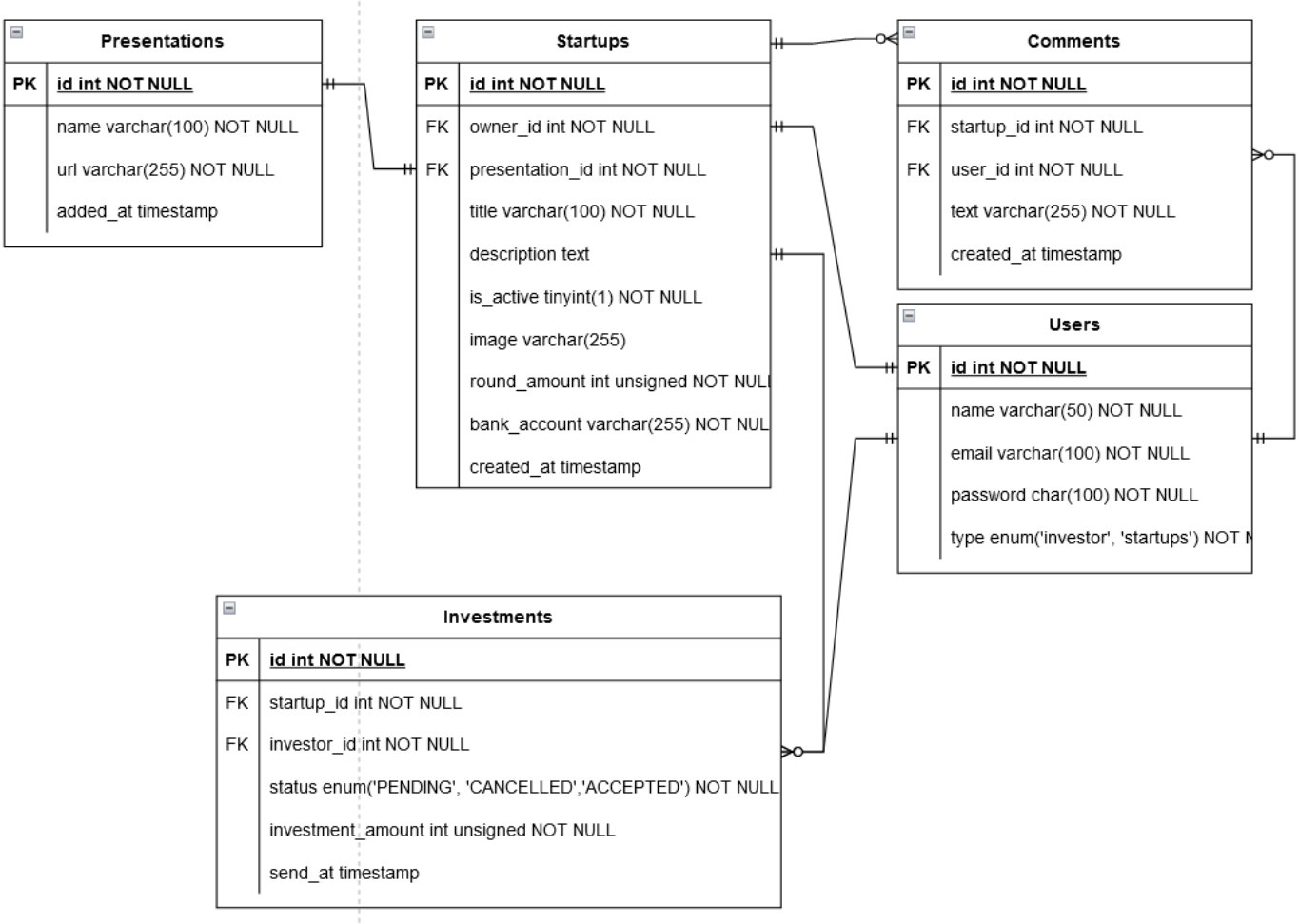
- Исключить повторяющие группы в отдельных таблицах.
- Создайте отдельную таблицу для каждого набора связанных данных.
- Определите каждый набор связанных данных с помощью основного ключа.

Вторая нормальная форма:

- Создайте отдельные таблицы для наборов значений, применимых к нескольким записям.
- Соотносим эти таблицы с иностранным ключом.

Предварительная схема нарушает свойство первой нормальной формы, так как в двух таблицах имеется повторяющееся поле **investment_amount.** Удалив данное поле в таблице **startups**, мы получим схему, соответствующую двум нормальным формам.

Получаем следующую схему:



7. SQL DDL

```
create table presentation
(
    id       int(10) unsigned auto_increment primary key not null,
    name     varchar(100)                                not null,
    url      varchar(255)                                not null,
    added_at timestamp                                   null
);

create table user
(
    id       int(10) unsigned auto_increment primary key not null,
    name     varchar(50)                                 not null,
    email    varchar(100)                                not null,
    password varchar(100)                                not null,
    type     enum ('investor', 'startups')               not null
);

create table startup
(
    id              int(10) unsigned auto_increment primary key not null,
    title           varchar(100)                                not null,
    description     text                                        null,
    is_active       tinyint(1)                                  not null,
    image           varchar(255)                                null,
    round_amount    int unsigned                                not null,
    bank_account    varchar(255)                                not null,
    created_at      timestamp                                   null,
    owner_id        int(10) unsigned                            not null,
    presentation_id int(10) unsigned                            not null,
    foreign key (owner_id) references user (id),
    foreign key (presentation_id) references presentation (id)
);

create table investment
(
    id                int(10) unsigned auto_increment primary key not null,
    investment_amount int unsigned                                not null,
    status            enum ('PENDING', 'CANCELLED', 'ACCEPTED')   not null,
    send_at           timestamp                                   null,
    startup_id        int(10) unsigned                            not null,
    investor_id       int(10) unsigned                            not null,
    foreign key (investor_id) references user (id),
    foreign key (startup_id) references startup (id)
);

create table comment
(
    id         int(10) unsigned auto_increment primary key not null,
    text       varchar(255)                                not null,
    created_at timestamp                                   null,
    startup_id int(10) unsigned                            not null,
    user_id    int(10) unsigned                            not null,
    foreign key (user_id) references user (id),
    foreign key (startup_id) references startup (id)
);
```

8. Запросы SQL DML

- Получение списка стартапов и их владельцев:

```
select startup.title, u.name
from startup
         left outer join user u on startup.owner_id = u.id;
```

- Получение списка инвесторов:

```
select count(*), *
from user
where type = 'investor';
```

- Получение списка инвесторов для стартапа:

```
select s.title, u.name, investment_amount
from investment
         left outer join user u on u.id = investment.investor_id
         left outer join startup s on s.id = investment.startup_id
where s.id = 1;
```

- Получение списка подтвержденных инвестиции и их инвесторов:

```
select u.name, investment_amount
from investment
         left outer join user u on u.id = investment.investor_id
where status = 'ACCEPTED';
```

- Получение списка всех инвестиции и их инвесторов:

```
select investment_amount, u.name
from investment
         left join user u on u.id = investment.investor_id;
```

- Получение количества инвестиции для определенного инвестора:

```
select count(investment_amount), u.name
from investment
         left join user u on u.id = investment.investor_id
where u.name = 'John';
```

- Получение списка инвесторов, сортированного по количеству инвестиции в обратном порядке:

```
select count(investment_amount) as investment_count, u.name
from investment
         left outer join user u on u.id = investment.investor_id
group by u.name
order by investment_count desc;
```

- Получение топ 10 инвесторов для стартапа по количеству переданных инвестиции:

```
select investment_amount as invest, u.name, s.title
from investment
         left outer join startup s on s.id = investment.startup_id
         left outer join user u on u.id = investment.investor_id
where s.id = 1
group by u.name
order by count(investment_amount)
limit 10;
```

9. Транзакции

- Подтверждение передачи инвестиции:

```
START TRANSACTION;
select *
from investment for
update;
update investment
set investment.status = 'ACCEPTED'
where investment.id = $id;
commit;
rollback;
```

- Деактивация стартапа:

```
start transaction;
select *
from startup for
update;
update startup
set is_active = 0
where id = $id;
commit;
rollback;
```